REMARKS

Claims 1-13, 15-25 and 37 remain pending in the application. Claims 1 and 21-23 have been amended without introduction of new matter. Favorable reconsideration is respectfully requested in view of the above amendments and the following remarks.

Claims 1-13, 15-25 and 37 stand rejected under 35 U.S.C. § 112, second paragraph, as allegedly being indefinite. In particular, the Office objected to the phrases "the whole" and "whenever" as allegedly being indefinite, and suggested that the word "whenever" should be replaced by "when". In order to expedite favorable prosecution of the application, claims 1 and 21-23 have been amended to address the Office's concerns. Accordingly, it is respectfully requested that the rejection of claims 1-13, 15-25 and 37 under the second paragraph of 35 U.S.C. § 112 be withdrawn.

Claims 1-13, 15-25, and 37 again stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Greene (U.S. Patent No. 6,631,419 – henceforth "Greene") in view of Wilkinson III et al. (U.S. Patent No. 6,014,659 – henceforth "Wilkinson"). This rejection is respectfully traversed.

The Greene and Wilkinson documents have remained throughout the present proceedings and, on various occasions, both Applicant and Examiner have expressed their opinions on the relevance or otherwise of the disclosure in, and teaching of, Greene and Wilkinson, singly or in combination. Applicant's earlier-filed arguments are hereby incorporated herein by reference. Thus, there is no need to reiterate Applicant's understanding of the teaching in Greene, other than to say that columns 7 and 8 of Greene describe a search engine using a pipelined process involving serialized operations, the engine functioning according to longest-prefix matching or masked-prefix matching principles.

The first 16 bits of a search key are used to search the first portion of a table so as to return a first next-hop value or a pointer to a second portion. A selector (130, Fig 1) selects between the returned first next-hop value and a next-hop value derived from the next 6 bits of the original key, the selector thereby generating a correct next-hop value that passes via a delay 126 to an output stage 128. If a pointer is returned from the first search, it is used in combination with the next 6 bits of the original key to generate, in address generator 112, a search address to search the second portion. That search either returns a second next-hop value, which passes to a second delay 132 at the output stage 128, or returns another pointer to a third table portion. That pointer is combined with a portion of the first pointer and the last 10 bits of the original key in address generator 114 to generate an address for searching

the third table portion. The result from the third portion is another next-hop value that is used in the output stage 128 to produce a final next-hop value. The overall process is described as "pipelined" in column 4, line 53 of Greene.

Where Greene differs fundamentally from Applicant's search engine is the pipelined nature and the consequential manner in which searches have to be conducted. The serialized processor in Greene (and also in Wilkinson) is in fact a thoroughly well-known way of increasing the performance of a serial operation by splitting it into stages. Pipelined (or serialized) search engines suffer from the defect that they operate like a production line, in that every item undergoes the same sequence of operations **in succession**. Not all searches take the same amount of time, however. The overall rate of performing operations is determined by the slowest stage and the outputs arrive in the same order as the inputs. In contrast, in a truly parallel implementation (such as in Applicant's search engine) the different search requests emerge "out of order". This is nevertheless much more efficient because as soon as any of the parallel state machines becomes free it (or they) can start a new search, rather than standing idle while the previous search completes (i.e., reaches the end of the whole of the serial process).

In Greene, as soon as the first stage of lookup has completed and the data has been passed on to the second stage, the first stage becomes free to accept a second request. It is not until this point that the first stage can actually accept any further requests. However, this second request is, necessarily, queued up behind the previous one. So although the processing is "overlapped" it is not truly concurrent, as specified in Applicant's invention.

It is not possible for the Greene search engine to commence a new search while the previous search is incomplete and is still progressing through the pipeline. To be more specific, Applicant submits that it is not possible for the first 16 bits of a new search key to enter the first search stage as the previous search enters the second or third stages because the results of the search in the first stage are employed in the second and subsequent search stages, as discussed above. It is to be remembered that returned data from the first search is passed to the output stage or to the address generator of the second and third stage.

It is therefore respectfully submitted that Greene does not in fact disclose or teach a search engine capable of performing multiple concurrent independent searches in the same table. Although Greene discloses conducting partial searches in portions of a table, these partial searches are themselves pipelined. Moreover, different searches are conducted in

pipelined fashion as discussed. In Applicant's respectful opinion, therefore, Greene does not provide a starting point for the present ground of rejection.

Although the Office states that "search engines are notoriously well known to comprise multiple state machines to handle multitasking", Applicant's experience is that although multiple state machines are often employed in search engines, it is not the case that multiple state machines are employed *to handle multitasking*. If the Office has evidence to support the view expressed in this Office Action, Applicant would be grateful for further information in order to address the objection if it should still prove necessary to do so, following this response.

Turning to Wilkinson, as previously discussed, Wilkinson discloses a specific form of a trie searching procedure in which nodes are selectively eliminated in order to reduce memory storage. Considering the router illustrated in Figure 3 of Wilkinson and the relevant description in columns 7-8 and 15, the address portion of an incoming packet is used in the search unit 40 of recognition engine 20 to address memory 50 containing destination data. The mechanism for obtaining the destination data is described in column 8, lines 2-10. It is clear from this that the search process is sequential in that a succession of intermediate results is necessary to determine the final result. In this regard, Wilkinson operates in the same manner as Greene. It follows that Wilkinson, like Greene, is only capable of handling a *single* request at any one time. The only relevant reference is in column 15, lines 22-25, where Wilkinson refers to "a number of registers and elementary state machines operating concurrently". However, this does not state that the state machines operate in parallel to enable multiple lookups of the same lookup table to be carried out concurrently, as stated in Applicant's claims.

Wilkinson has apparently been cited merely as an example of a search engine using multiple state machines. As Applicant has previously argued, Wilkinson may use multiple state machines but the result is that Wilkinson is still incapable of searching more than a single request at a time. The mere inclusion of a plurality of state machines does not inevitably lead to the table being accessed simultaneously in respect of a plurality of *different* search requests. It is respectfully submitted that carrying out sequential partial searches in respect of a single search request, as in Wilkinson, is not the same as carrying out concurrent, parallel searches throughout the table in respect of different search requests.

Applicant therefore concludes that Wilkinson, like Greene, does not disclose or teach multiple, independent, concurrent searches. It is respectfully maintained that there is therefore no incentive or motivation to combine these documents in the manner suggested by the Office. Moreover, even such a combination does not teach the combination of features claimed in the current set of claims.

Applicant notes the Office's argument that Greene, in column 8, lines 30-35, states that "several queries are processed simultaneously" as a result of a DEST-IP value being applied at the input as soon as one DEST-IP value has been used to generate an address for the first memory array. However, it should be recognized that *these operations are still sequential* in that one DEST-IP value cannot enter the search engine until the previous DEST-IP has itself moved on to search the second array. The Greene search engine is inherently incapable of performing a number of concurrent, independent search requests, unlike Applicant's search engine, where a plurality of independent search requests can arrive at the search engine simultaneously, as described in page 10, lines 5-8 and 30-37.

Applicant also repeats here the previous observation that Applicant's multiple state machines are used to accelerate the *overall* look-up process, where the ability to look up several values in parallel means that *all* of the look-ups can be completed faster (i.e., the latency for any given batch of search requests is reduced). In Applicant's invention, all of the finite state machines are able to reference the whole of the lookup table data at the same time whereas in Greene the state machine can only reference one region of the array at a time. Relevant passages in pages 4, 11, 12 and 14 have already been indicated in the previous response and will not be repeated here but the Office is invited to look at those passages once more to appreciate the difference between Applicant's invention and the teaching in Greene and Wilkinson. In Applicant's invention, all the state machines have access to all the memory banks all the time via the distributor block 204, so multiple independent lookups can be done concurrently, with each state machine having access to all the data in the whole of the memory.

In a further attempt to specify as clearly as possible the features that contribute to the inventiveness of Applicant's invention, claims 1, 21, 22 and 23 have each been further amended to emphasize that the look up engine comprises a plurality of look up state machines connected in parallel to enable multiple look ups of <u>different search requests to be carried out in the same look up table concurrently</u>, the state machines all having concurrent access to all the entries in the look up table when they perform a look up. (Emphasis added.)

This amendment also avoids the need to resolve any questions as to whether a single table is the same as a table divided into memory banks and whether multiple memory banks can be considered as a single table.

For at least the foregoing reasons, it is respectfully asserted that independent claims 1, 21, 22, and 23, as well as their dependent claims 2-13, 15-20, 24-25 and 37 are patentably distinguishable over the prior art of record. Therefore, it is respectfully requested that the rejection of these claims under 35 U.S.C. §103(a) be withdrawn.

The application is believed to be in condition for allowance. Prompt notice of same is respectfully requested.

Respectfully submitted,

Potomac Patent Group PLLC

Date: March 18, 2007

By: /Kenneth B. Leffler, Reg. No. 36,075/
Kenneth B. Leffler
Registration No. 36,075

P.O. Box 270
Fredericksburg, Virginia 22404
703-718-8884